Maxima quick reference guide

All Maxima commands should end with either ; (which causes any output produced to be displayed) or $ (which suppresses output).

Mathematical operations and functions

	Example
Addition	`4+3;`
Subtraction	`4-3;`
Multiplication	`4*3;`
Division	`4/3;`
Brackets	`2*(3+4);`
Powers	`2^3;` *or* `2**3;`
Square root, $\sqrt{x}$	`sqrt(5);`
Exponential, e^x	`%e^3;` *or* `exp(3);`
Natural logarithm, ln	`log(8);`
Magnitude/modulus, $\|x\|$	`abs(-3); abs(2+%i);`
Equality (in an equation)	`2*x+3=5;`

sin, cos, tan	`sin(%pi); cos(%pi/2); tan(%pi/4);`
cosec , sec, cot	`csc(%pi/3); sec(%pi/3); cot(%pi/4);`
$\sin^{-1}$, $\cos^{-1}$, $\tan^{-1}$	`asin(1/2); acos(1/2); atan(1/2);`

The arguments of trigonometric functions are in radians.

sinh, cosh, tanh	`sinh(0); cosh(0); tanh(0);`
cosech , sech , coth	`csch(1); sech(0); coth(1);`
$\sinh^{-1}$, $\cosh^{-1}$, $\tanh^{-1}$	`asinh(0); acosh(1); atanh(0);`

Factorial, $n!$	`5!;` *or* `factorial(5);`
Binomial coefficient, ${}^nC_r$	`binomial(5,2);`
Highest common factor (greatest common divisor)	`gcd(6,4);`

Quotient and remainder on dividing numbers or polynomials	`quotient(x^2,x-1);` *and* `remainder(x^2,x-1);` (*or* `divide(x^2,x-1);`)

Round to nearest integer	`round(2.4);`
Round to integer below	`floor(2.4);`
Round to integer above	`ceiling(2.4);`

Limit of expression (e.g. $\lim_{x\to 0} x^2$)	`limit(x^2,x,0);`

Mathematical constants and symbols

Constant	*Syntax*	*Symbol*	*Syntax*
e	`%e`	∞	inf
π	`%pi`		
i	`%i`		

Help

	Example
Get help on a command	`? float;` *or* `describe(float);`
Find help information whose title contains the given text	`?? flo;` *or* `describe(flo,inexact);`

Useful commands

	Example
Output of the previous command	`%`
Output of an earlier command (e.g. line 2)	`%o2`
Convert to a decimal number	`float(sqrt(2));` *or* `sqrt(2),numer;`
Set number of significant figures displayed	`fpprintprec:3;`
Load an additional package	`load(implicit_plot);`
Reset all system variables	`reset();`
Prevent the evaluation of a command	`'diff(y,x)`
Force the evaluation of a command	`f(x):=''(diff(x^4,x));`

Algebraic manipulation

	Example
Expand brackets	`expand((x+1)^2);`
Factorise	`factor(2*x+4*x^2);`
Simplify a rational expression	`fullratsimp((2*x+4*x^2)/x);`
Express in partial fractions	`partfrac(2/(x^2-1),x);`
Numerator	`num(x/(x+1));`
Denominator	`denom(x/(x+1));`
Simplify an expression involving exponentials or logarithms	`radcan(log(x^2*y));`
Combine logarithms	`logcontract(log(a)+log(b));`
Expand trigonometric functions of sums and multiples of angles	`trigexpand(sin(A+B));` *or* `sin(A+B),trigexpand;`
Reduce powers of trig functions to functions of multiples of angles	`trigreduce(sin(x)^2);`
Simplify using trig identities	`trigsimp(sin(x)^2+cos(x)^2);`
Simplify a rational trig expression	`trigrat(sin(2*x)/cos(x));`
Left-hand side of an equation	`lhs(4*x+1=2*x-2);`
Right-hand side of an equation	`rhs(4*x+1=2*x-2);`
Substitute (e.g. 4 for `x` in `x^2`)	`subst(4,x,x^2);` *or* `x^2,[x=4]`

Variables and functions

	Example
Assign a value to a variable	`a:23;`
Display the value of a variable	`a;`
List user-defined variables	`values;`
Define a function	`f(x):=2*x+3;`
Evaluate a function at a value	`f(1);`
List user-defined functions	`functions;`
Remove an assigned variable or function	`kill(a); kill(f);`
Remove all assigned variables and functions	`kill(all);`

Assumptions about variables

	Example
Make an assumption about a variable	`assume(a>0);`
State that two things are equal	`assume(equal(n,3));`
State that two things are not equal	`assume(notequal(n,-1));`
Forget an assumption	`forget(a>0);`
Declare a property of a variable	`declare(n,integer);`
Remove a property from a variable	`remove(n,integer);`
List all known facts	`facts();`
List all known facts about a variable	`facts(a);`

Lists

List are given within square brackets ([,]), with elements separated by commas.

	Example
Assign a list to a variable	`A:[1,3,5,7,11];`
Length of a list	`length(A);`
Element of a list (e.g. 2nd)	`A[2];`
First, second, ..., tenth element	`first(A); second(A); ..., tenth(A);`
Last element	`last(A);`

Create a list using a general term, for example:

general term $2n$, with n from 1 to 100	`makelist(2*n,n,1,100);`
general term $2n$, with n from 1 to 100 in steps of 2	`makelist(2*n,n,1,100,2);`
general term $2n$, with n taken to be each element of the list `A`	`makelist(2*n,n,A);`

Vectors

	Example
Assign a vector (e.g. $\mathbf{i}+2\mathbf{j}+3\mathbf{k}$) to a variable	`a:[1,2,3];`
Vector addition	`a+b;`
Vector subtraction	`a-b;`
Multiplication by a scalar	`2*a;`
Scalar (dot) product	`a.b;`
Vector (cross) product	`express(a~b);`

Before using this, load the necessary package using `load(vect);`

Matrices

	Example
Assign a matrix (e.g. $\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}$) to a variable	`A:matrix([1,2],[3,4]);`
Size of a matrix	`matrix_size(A);`
Row of a matrix (e.g. 2nd)	`A[2];`
Element of a matrix	`A[3,4];` *or* `A[3][4];`
Matrix addition	`A+B;`
Matrix subtraction	`A-B;`
Multiplication by a scalar	`2*A;`
Matrix multiplication	`A.B;`
Matrix power	`A^^3;`
Transpose of a matrix	`transpose(A);`
Determinant of a square matrix	`determinant(A);`
Inverse of a square matrix	`invert(A);` *or* `A^^(-1);`
Identity matrix (e.g. of size 2×2)	`ident(2);`
Characteristic polynomial of a matrix (e.g. expressed in terms of the variable `x`)	`charpoly(A,x);`
Eigenvalues of a matrix	`eigenvalues(A);`

The first element of the list output contains the eigenvalues, and the second their multiplicities.

Eigenvectors of a matrix	`eigenvectors(A);`

The first element of the list output contains the eigenvalues and their multiplicities, the second element contains the eigenvectors.

Version 1.0
© 2016 The Open University

Sequences and series

	Example
Define a sequence using a closed form	`a[n]:=n^2;`
Define a sequence by recurrence	`b[1]:1;` `b[n]:=2*b[n-1];`
Term of a sequence	`b[50];`
Sum of a series	`sum(n^2,n,1,10);`
Permit simplification of summations	`simpsum:true;`

Complex numbers

	Example
Real part of a complex number	`realpart(1+2*%i);`
Imaginary part of a complex number	`imagpart(1+2*%i);`
Conjugate of a complex number	`conjugate(1+2*%i);`
Argument of a complex number	`carg(1+2*%i);`
Cartesian form of a complex number	`rectform(2/%i);`

Solving equations

	Example
Solve an equation, analytically	`solve(2*x^2-1=0);`
Solve an equation for a variable	`solve(2*a*b-3*b=0, a);`
Solve simultaneous equations	`solve([x+y=4,4*x-y=1],[x,y]);`
Solve simultaneous linear equations	`linsolve([x+y=4,4*x-y=1],[x,y]);`
Numerically find a root within a given interval	`find_root(2*x^2-1,x,0,1);`
Numerically find all roots of a polynomial	`allroots(2*x^3-1=0);`

Solve a recurrence system `solve_rec(a[n]=2*a[n-1],a[n],a[1]=1);`
Before using `solve_rec` *load the necessary package using* `load(solve_rec);`

Calculus

	Example
Differentiate	`diff(sin(x^2),x);`
Differentiate twice	`diff(log(x),x,2);`
Integrate	`integrate(x^2,x);`
Evaluate a definite integral	`integrate(sin(x),x,0,1);`
Numerically approximate a definite integral	`quad_qags(exp(-x^2),x,0,1);`

The output lists: the value, the estimated error of the approximation, the number of integrand evaluations required, an error code.

Find a Taylor polynomial (e.g. for $\sin x$, about $x = 0$, cubic)	`taylor(sin(x),x,0,3);`

Fourier series

Before using these commands, load the package using `load(fourie);`

	Example
Calculate Fourier coefficients (e.g. of x^2 over $-\pi \le x \le \pi$)	`coeff:fourier(x^2,x,%pi);`
Use coefficients to find a Fourier approximation (e.g. of order 4)	`fourexpand(coeff,x,%pi,4);`

Graph plotting (using `plot` commands)

Commands beginning with `wx` are appropriate for wxMaxima.
In other interfaces, use equivalent commands without the `wx` prefix.

	Example
Plot the graph of a function over an interval	`wxplot2d(x^2, [x,0,1]);`
Plot the graphs of several functions over an interval	`wxplot2d([x^2, 2*x], [x,0,1]);`
Plot a graph from a set of points on it	`wxplot2d([discrete,[[1,5],[2,10]]]);`
Plot a set of points	`wxplot2d([discrete,[[1,5],[2,10]]],[style, points]);`
Plot a curve from parametric equations	`wxplot2d([parametric, 2*t, 3*t+1, [t,0,1]]);`
Plot a curve from an equation in implicit form	`wximplicit_plot(x^2+y^2=1,[x,-1,1],[y,-1,1]);`
Plot several curves from equations in implicit form	`wximplicit_plot([x^2+2*y^2=4,y=x^3],[x,-1,1],[y,-1,1]);`
Before using `wximplicit_plot` *load the necessary package using*	`load(implicit_plot);`
Plot contours of a function of two variables	`wxcontour_plot(x*y,[x,-1,1],[y,-1,1]);`
Plot a surface from a function of two variables in three dimensions	`wxplot3d(x*y,[x,0,1],[y,0,1]);`
(In wxMaxima only) Change the size of graphs (e.g. to 800×600 pixels)	`wxplot_size:[800,600];`

Plotting options

These options can be included as additional arguments of the commands above.
To set an option as a default option for subsequent plotting commands, use `set_plot_option(`*option*`);`

	Example option
Vertical range	`[y, 0, 5]`
Curve colours	`[color, red, green]`
Legend text	`[legend, "f(x)","g(x)"]`
Turn legend off	`[legend, false]`
Position legend at the bottom of the graph	`[gnuplot_preamble,"set key bottom"]`
Horizontal axis label	`[xlabel, "time, t (h)"]`
Vertical axis label	`[ylabel, "displacement, s (km)"]`
Plot points, not curves	`[style, points]`
Plot first graph using points, second as a curve	`[style, points, lines]`
Number of points for parametric curves	`[nticks, 100]`

Plot with equal axis scales	`[gnuplot_preamble,"set size ratio -1"]`
	(For use with older versions of Maxima)
	`[same_xy,true]`
	(For use with newer versions of Maxima)

Change the number of function contours plotted	`[gnuplot_preamble,"set cntrparam levels 10"]`

Graph plotting (using `draw` commands)

Commands beginning with `wx` are appropriate for wxMaxima.
In other interfaces, use equivalent commands without the `wx` prefix.
Before using these commands, load the package using `load(draw);`
Draw in 2D `wxdraw2d(`*objects*`,`*options*`);`
Draw in 3D `wxdraw3d(`*objects*`,`*options*`);`
The *objects* and *options* are taken from the lists below.

Objects (2D)

(Objects in 3D are similar but also require ranges of y and/or z.)

	Example
Curve from an explicit equation	`explicit(x^2,x,0,1)`
Curve from an implicit equation	`implicit(x^2+y^2=1,x,-1,1,y,-1,1)`
Ellipse (e.g. centre (0,0), radii 1,2)	`ellipse(0,0,1,2,0,360)`
Rectangle	`rectangle([0,0],[1,2])`

Drawing options

	Example
Horizontal range	`xrange=[0,3]`
Vertical range	`yrange=[0,5]`
Colour of following objects	`color=green`
Colour to fill following objects	`fill_color=blue`
Plot with equal axis scales	`proportional_axes=xy`

Differential equations

	Example
Solve a differential equation analytically	`sol:ode2('diff(y,x)=x,y,x);`
Apply one initial condition to the analytic solution (e.g. $y(0) = 1$)	`ic1(sol,x=0,y=1);`
Apply two initial conditions to the analytic solution (e.g. $y(0) = 1$, $\frac{dy}{dx}(0) = 0$)	`ic2(sol,x=0,y=1,'diff(y,x)=0);`
Apply two boundary conditions to the analytic solution (e.g. $y(0) = 1$, $y(1) = 2$)	`bc2(sol,x=0,y=1,x=1,y=2);`
Solve a differential equation numerically (e.g. $\frac{dy}{dx} = x^2$ with $y(0) = 1$ in $0 \le x \le 10$, with step size of 0.1)	`rk(x^2,y,1,[x,0,10,0.1]);`

Systems of differential equations

	Example
Solve a system of differential equations analytically	`desolve([diff(x(t),t)=2*y, diff(y(t),t)=3*x],[x(t),y(t)]);`
Specify an initial condition (e.g. $x(0) = 1$)	`atvalue(x(t),t=0,1);`
List all initial conditions	`printprops(all,atvalue);`
Remove an initial condition (e.g. for x)	`kill(x);`

Plotting direction fields

Plot a direction field `wxdrawdf(x*y,[x,y],field_arrows=false);`
Before using this load the necessary package using `load(drawdf);`

Direction field plotting options

These (and the objects for `draw`) can be included as additional arguments of the `wxdrawdf` command.

	Example
Range of a variable	`[x,0,5]`
Add the graph of a particular solution passing through a given point	`[trajectory_at, 0, 1]` *or* `soln_at(0,1)`
The number of steps used when approximating a particular solution	`[nsteps, 200]`
Mark a point (e.g. at $(0, 0)$)	`point_at(0,0)`